EXTRAHOP

WHITE PAPER

Don’t Leave Your SOC Agents Starving for Context

Why Agentic SOC Success Depends on Network Telemetry

TABLE OF CONTENTS

Agentic SOC implementations often focus on the model at its heart as the key to its success. This is understandable; the LLM provides the reasoning that guides agentic decision-making and actions. But this approach overlooks a crucial consideration: the accuracy and reliability of an agent's reasoning depend largely on the data it's provided. To understand the meaning of an event or alert, it has to know its full context—not just the assets in the environment, but also the relationships connecting them, the behaviors considered normal, and the privileges defining their reach. The totality of data makes the context layer the most important element of an agentic SOC. Unfortunately, it's often treated as an afterthought.

In this white paper, we'll explain why the context layer should be the first step in agentic SOC implementation, how network telemetry takes agents from information to insight, and what a ground-truth network layer looks like.

How the context gap sets a prototyping trap

For SOC leaders, agentic SOC implementation has become an urgent priority. Frontier models have collapsed the adversary's timeline, with both discovery and exploitation happening in a single working session. The traditional SOC workflow of queue, triage, investigate, and escalate assumes time defenders no longer have. Agentic AI can transform and accelerate SOC workflows to counter machine-speed threats, but only if it's put to work effectively. In the absence of a definitive implementation blueprint for building an agentic SOC, many organizations are improvising and prototyping as they go. This is understandable given the pressure—but it's unlikely to achieve optimal results.

On an architectural level, most early agentic SOC operating models encompassed five layers: Detection, Context, Agent, Action, and Human. More recently, agentic SOC leaders have defined a new operating model designed to give autonomous security agents the evidence, governance, and reasoning they need to act with precision. The new architecture comprises three operational layers with human oversight across the system as a whole:

Context	Harness	Model
The evidence agents reason on.	The governed control plane that decides how agents are allowed to act on that evidence, including guardrails, permissions, and audit trails.	The reasoning layer that interprets context and recommends action.

In both traditional five-layer SOC architecture and the new operating model, it's the context layer that answers the questions that a human analyst focuses on. What is this device? Which business process does it support? What does it normally communicate with? Has this behavior appeared before? What changed recently? If it's compromised, what would be the blast radius of the attack? Without this understanding, a human analyst wouldn't be able to accurately interpret an alert or event, prioritize the response, or take action to address the threat. The same is true for a SOC agent.

In spite of its central importance, the context layer is consistently underbuilt in agentic SOC implementations. Attention and investment are directed instead to the agent layer, including model selection, prompt engineering, and orchestration frameworks, with context treated as a secondary configuration task to be completed after more interesting engineering work is done. Often, this amounts to connecting a SIEM as the sole source of security information and events, but this is far from sufficient. In fact, leaving the context layer for last is the most common reason agentic SOC programs stall.

When organizations build model sophistication before context infrastructure, they fall into the Prototyping Trap: the system looks sound in a controlled prototype but fails in production. Agents produce fluent, confident-sounding output that builds trust in their accuracy, but this veneer of intelligence obscures the gaps in their inputs that impair their reasoning. That's worse than no automation, as it can lead analysts to approve or act on flawed output, leaving risk in the environment. When errors become apparent, human override rates climb and trust in the system erodes.

SIEMs and LLMs can't carry the context layer

Many teams trust that the SIEM can serve as an adequate context layer, and that a capable model will absorb whatever the SIEM leaves out. Both halves of that assumption are wrong, and together they explain why so many programs stall.

The SIEM is essential infrastructure for every mature SOC, traditional or agentic. It's unmatched at ingesting logs, normalizing fields, and enabling rapid queries across billions of events. What it can't do is provide an understanding of what assets mean, how systems relate, or what behavioral normal looks like for a specific device in a specific environment. Relying on a SIEM to provide context creates several problems:

Empty results are ambiguous.
When an agent queries the SIEM and gets zero rows back, it has no way to tell whether nothing happened or whether the query was wrong. A human analyst would know to be suspicious of a clean result. The agent reports "no indication of lateral movement, confidence high" and moves on.

Records fail to reflect criticality.
A SIEM can register that an IP made a connection, but it doesn't know that the IP belongs to the most sensitive production database in the environment. The record of that event lacks the information an agent needs to assess its significance, leaving it unable to make effective decisions—even if it doesn't admit this.

Every query starts from scratch.
A SIEM records events; it doesn't map the environment. It knows a log was written, but it doesn't know that this host is a domain controller normally talking to 40 peers and currently talking to 400. The agent has to rebuild that picture from raw queries on every turn, at a cost that scales with the volume of text rather than the value of the answer.

Silence and quiet look identical.
Everything in a SIEM was reported by something else, and that something can be disabled, wiped, or simply configured not to emit. An empty log stream may mean nothing happened—or may mean the attacker got there first and covered their tracks.

The shortcomings of a SIEM-only context layer carry over to the LLM that relies on it. As a reasoning engine, a model works with whatever data it's given and presents a result with the same assurance whether the input is complete or full of holes. Given inadequate data, even the most powerful AI will provide flawed output. Sometimes, it just makes things up.

Why network telemetry is non-negotiable

When SOC teams recognize that their system is producing errors and hallucinations, a natural response is to layer in more sources, such as CMDBs, endpoint agents, log pipelines, and identity systems. Each of these can provide valuable context, but each also carries a structural blind spot that can undermine agentic SOC performance.

CMDBs

Capture what IT intended the environment to look like, not what it is. Manually maintained and chronically out of date, they typically miss shadow IT, unmanaged devices, transient cloud workloads, and IoT assets. An agent reasoning over stale and incomplete CMDB data makes critical decisions about a fictional environment, leaving the real organization at risk.

Endpoint agents (EDR)

Have inherent coverage gaps. They can't deploy across IoT, OT, unmanaged devices, legacy systems, or many cloud workloads. The segments hardest to instrument are often the most attractive to attackers precisely because they carry less scrutiny.

Log pipelines

Drop events under heavy load and can be suppressed or rewritten by a sophisticated attacker who has gained access. An agent reasoning over records that are both incomplete and mutable can't be expected to produce reliable results.

Identity systems

Can't see past legitimate-looking credentials. A hijacked active session produces no failed login event to scrutinize, no MFA challenge to defeat, and no anomaly to flag. The identity system reports authenticated sessions, but it can't confirm that the session belongs to the credential's owner.

Network telemetry shares none of these blind spots, closing the coverage gaps left by other sources. Adversaries can delete local logs, spoof credentials, and evade endpoint agents, but the infrastructure records every packet that crosses it. Every lateral move, command-and-control callback, and attempt at exfiltration is captured with the details agents need for effective analysis, decisions, and action. The network never lies.

Six elements of a mature context layer

The context layer is a sequenced set of capabilities, each building on the one before it. Together, they give an agent the environmental understanding it needs to reason accurately.

1.

Live asset and topology intelligence

Topology intelligence is structural. It tells the agent what exists and how assets connect—the communication pathways, peering relationships, and protocol dependencies that define normal architecture. A mature context layer maintains a continuously updated inventory of every device, service, cloud workload, identity, and network dependency, derived from live telemetry rather than static records. By maintaining a real-time map of which systems talk to each other, over what protocols, and at what frequency, it enables the agent to judge whether a connection fits the established architectural pattern rather than merely noting that a connection occurred.

Topology intelligence anchors the context layer. Without an accurate, current picture of what's on the network and how it connects, no subsequent element is reliable.

2.

Behavioral baselines

Where topology intelligence is structural, behavioral baselines are temporal. They capture how assets actually behave across time, not just how they're connected. Rolling baselines for users, devices, applications, and services—built from observed traffic history—tell the agent what volumes are normal for this device at this hour, what destinations are typical, and what protocols usually appear in what sequence.

This temporal layer answers a different question from topology. Topology tells the agent whether a connection fits the architecture. Baselines tell the agent whether it fits the history. Without them, the agent cannot distinguish a novel threat from a scheduled maintenance window, or a legitimate architectural change from an attacker's lateral movement.

3.

Identity, privilege, and blast radius mapping

Before an agent recommends containment, it needs to know the potential extent of the breach. This requires mapping active identities, service accounts, permissions, and sessions to the assets they can access, modify, or disrupt. The agent also needs to understand the technical impact of any containment action—what systems would lose connectivity, what services depend on this asset, and what the downstream effects would be.

Accurate blast radius mapping depends on accurate topology intelligence. You can't assess the significance of a compromised asset if you don't know what it connects to.

4.

Business criticality and data sensitivity

The same alert can require fundamentally different response logic depending on whether it occurs on a development server or a production database holding regulated customer data. Business criticality context determines not just whether to act but also how aggressively—and what level of operational disruption is acceptable given the specific asset at risk.

This element of context is frequently missing from automated programs because it can't be derived from telemetry. Mapping assets to business function and regulatory context requires deliberate input from asset owners and compliance teams. Otherwise, agents apply identical response logic to assets with radically different operational significance.

5.

Change and configuration history

Anomalous behavior following a deployment is common and usually benign. Some of it comes from the deployment process itself, including services restarting, packages being pulled, and agents checking in. The rest results from what was deployed and how it changed ongoing operations, such as a new service communicating on different ports, altered traffic volumes from a configuration change, or a dependency resolving to a different endpoint. Without change context, the agent can't tell either apart from a genuine attack. With it, the agent can correlate the timing, scope, and nature of what it's seeing against known operational activity before escalating.

Agents depend on other elements of the context layer to interpret change data accurately. Baselines signal that current behavior deviates from what was normal before, while topology intelligence tells the agent whether the relevant assets were touched by a recent deployment.

6.

Enriched threat intelligence

A threat intel feed tells you which indicators are dangerous somewhere, but it doesn't tell you which ones matter to your environment. Without network context, the agent has no way to work out the difference.

Network telemetry closes that gap. A threat intel entry becomes operationally significant only when your assets are actually communicating with the flagged infrastructure, a communication that only the network sees in real time.

Connecting the context dependency chain

The sequence of the elements above matters. The earliest are structural dependencies: you can't build meaningful baselines without an accurate asset inventory, and identity and blast-radius mapping fall apart without baselines to establish what normal access looks like. The later elements—change context and threat intelligence—can be assembled independently, but they don't produce reliable output on their own. Correlating external indicators against an environment the agent doesn't understand produces analysis that sounds authoritative but isn't. Programs that build backward, assembling threat intel pipelines or LLM orchestration before a reliable asset baseline, consistently land in the Prototyping Trap.

Agents need context as a real-time service

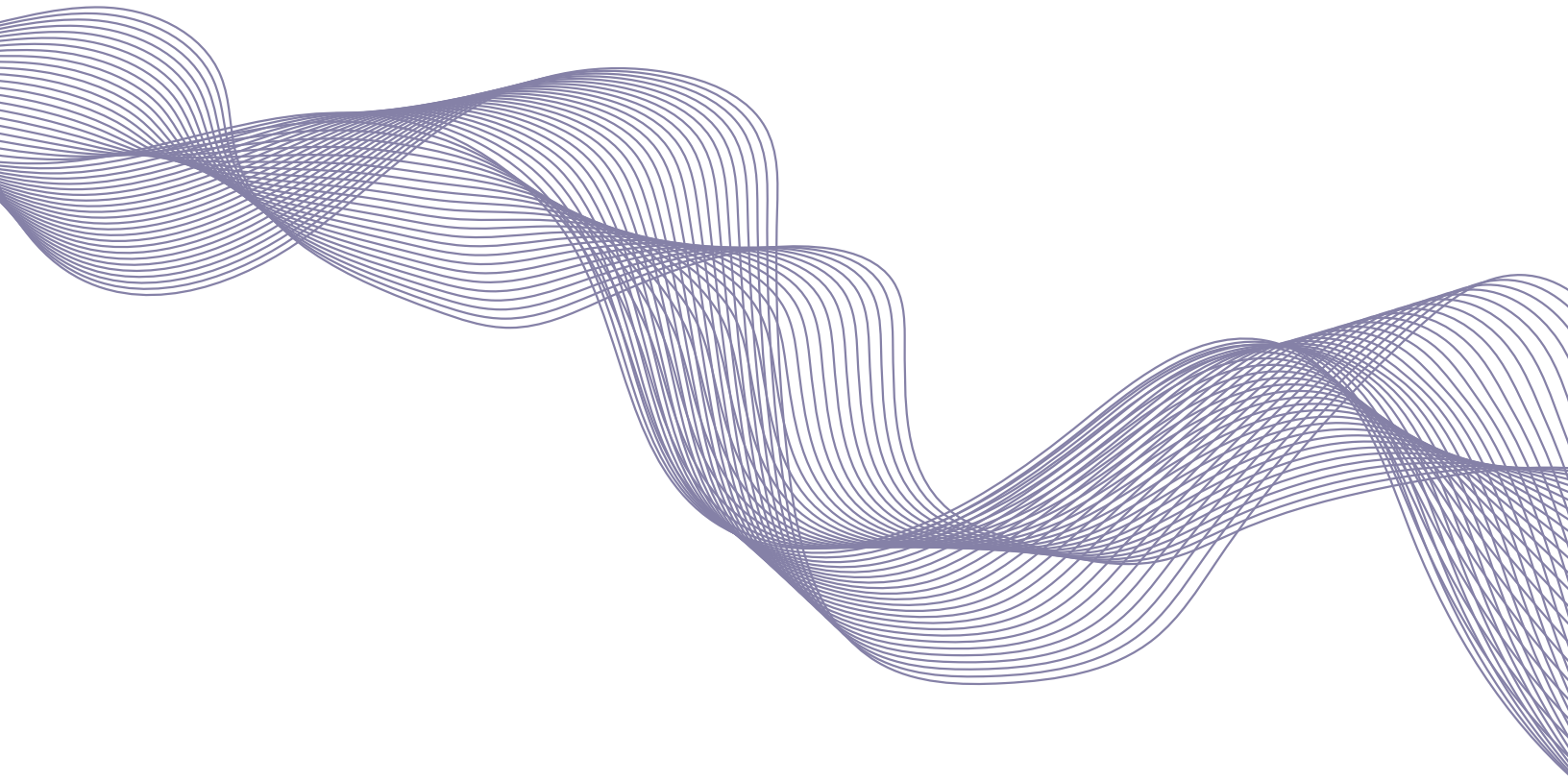
An agent investigating a live alert can't wait for a nightly CMDB sync or a batch log export. High-fidelity decisions require contextual answers in seconds, and that requirement shapes two aspects of the context layer: how the agent retrieves context, and how fresh that context actually is.

For fast retrieval, the context layer needs to reach agents through a real-time context API—an orchestration gateway that aggregates signals from network detection and response (NDR) and endpoint detection and response (EDR) tools, the configuration management database (CMDB), identity providers, and threat intel, normalizes them into a unified schema, and delivers structured responses on demand. Without that unifying layer, agents make sequential, slow, multi-system queries, and the context they assemble is already outdated by the time the investigation concludes.

The need for fresh data is often underestimated by agentic SOC programs. A device classification accurate at 9 a.m. may be wrong by noon if a workload migrated, a container spun up, or an adversary moved laterally and established persistence on a new host. A fast API that pulls from stale sources returns quick, confident answers about an environment that no longer exists.

The need for fresh data is often underestimated by agentic SOC programs.

For both of these reasons, the context layer is anchored by the NDR platform. It answers agent queries at machine speed, and its underlying data refreshes automatically from live traffic—continuously discovering assets, observing protocols, building behavioral baselines, and surfacing lateral movement indicators without manual entry or batch dependency. Other sources enrich that record, but they don't replace it. NDR is the only source in the pipeline whose refresh cycle matches the speed of the threats the agent is investigating.



What a ground-truth network layer looks like

The agentic SOC operating model the industry aligns around isn't going to be delivered by any single vendor. That's why ExtraHop helped launch the Agentic SOC Alliance—a group of network detection, endpoint, identity, threat intelligence, AI-native SOC platform, orchestration, and adversarial validation vendors aligned on a shared architecture for autonomous security operations. The Alliance provides architectural leadership and guidance to help security leaders assemble an agentic SOC out of best-of-breed capabilities.

Within the Context-Harness-Model operating model, ExtraHop provides the real-time Context layer—the ground-truth network evidence agents reason over. RevealX delivers passive, continuous, full-fidelity network analysis across cloud, on-premises, and hybrid environments, integrated through a real-time API that agent orchestration frameworks can query at machine speed. The wire data records RevealX generates are purpose-structured for AI consumption: normalized, API-queryable, and returned in milliseconds. This eliminates the token cost and latency problems that can make raw packet inspection impractical for agentic workflows.

From network traffic alone, ExtraHop supplies the context an agent needs first:

Live asset and topology intelligence
continuously discovered from observed communications, with no manual entry and no coverage gaps

Behavioral baselines
built from wire data and updated in real time as the environment evolves

Identity context
derived from observed sessions and protocol behavior, including lateral movement indicators that bypass authentication logging

Detection across unmanaged, IoT, and OT segments
that endpoint agents can't reach

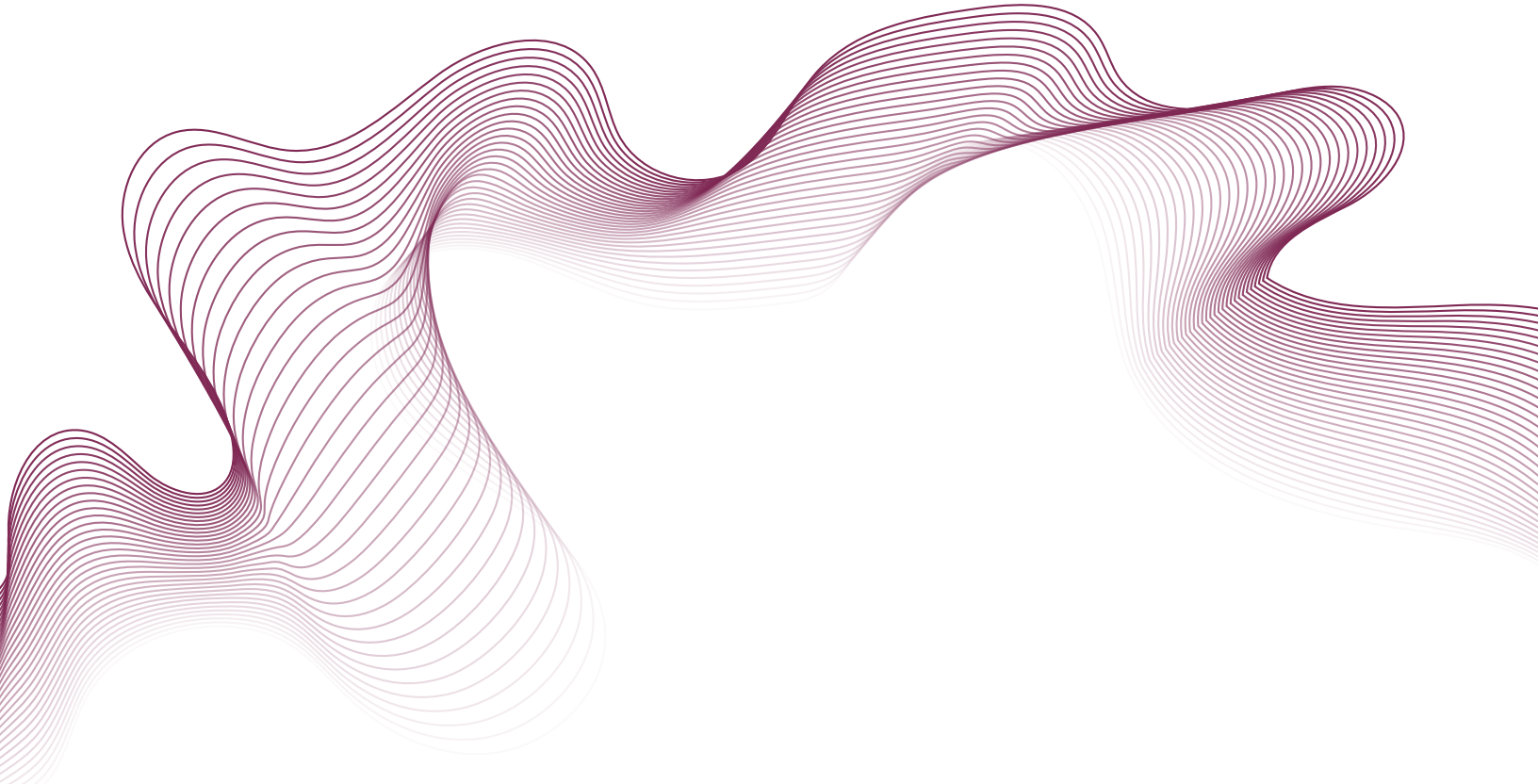
While business criticality and data sensitivity are organizational judgments beyond the scope of network telemetry, ExtraHop integrates with the asset tagging, compliance scoping, and CMDB systems where these classifications are defined, so the agent reasons over organizational context alongside network context.

With ExtraHop, SOC agents reason over a context layer that is authoritative, continuously updated, and grounded in evidence adversaries can't avoid generating.

The discipline behind working agentic SOC programs

The LLM that powers your agents matters—but the quality of your context layer matters even more. By building your data infrastructure first, with network telemetry as its foundation, you enable fast, accurate agent decisions that gain the trust of analysts, allow you to expand and scale your program, and free experienced analysts to focus on creative adversarial thinking rather than data gathering.

Learn more about the **new operating model** that is transforming the Agentic SOC.



ABOUT EXTRAHOP

ExtraHop is a leader in real-time network intelligence, giving organizations the high-fidelity context to power security and IT AI automation, detect risks faster, and respond with confidence. AI agents rely on ExtraHop's real-time, ground-truth data to operate reliably in the SOC and NOC. For security teams, that means surfacing threats and definitive evidence to investigate them at machine speed. For IT operations, it means resolving performance issues in seconds. Engineered for unmatched scale, the ExtraHop RevealX platform delivers a unified view of the network across data centers, campus, branch, cloud, and AI environments.

To learn more, visit extrahop.com or follow us on LinkedIn.

EXTRAHOP®

info@extrahop.com
extrahop.com